ViFin: Harness Passive Vibration to Continuous Micro Finger Writing with a Commodity Smartwatch

WENQIANG CHEN, University of Virginia
LIN CHEN, VibInt AI Limited
MEIYI MA, University of Virginia
FARSHID SALEMI PARIZI, University of Washington
SHWETAK PATEL, University of Washington
JOHN STANKOVIC, University of Virginia

Wearable devices, such as smartwatches and head-mounted devices (HMD), demand new input devices for a natural, subtle, and easy-to-use way to input commands and text. In this paper, we propose and investigate ViFin, a new technique for input commands and text entry, which harness finger movement induced vibration to track continuous micro finger-level writing with a commodity smartwatch. Inspired by the recurrent neural aligner and transfer learning, ViFin recognizes continuous finger writing, works across different users, and achieves an accuracy of 90% and 91% for recognizing numbers and letters, respectively. We quantify our approach's accuracy through real-time system experiments in different arm positions, writing speeds, and smartwatch position displacements. Finally, a real-time writing system and two user studies on real-world tasks are implemented and assessed.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *wearable devices*; vibration intelligence; • **Text input** → micro finger writing.

Additional Key Words and Phrases: micro finger writing, text input, wearable devices, vibration intelligence

ACM Reference Format:
Wenqiang Chen, Lin Chen, Meiyi Ma, Farshid Salemi Parizi, Shwetak Patel, and John Stankovic. 2021. ViFin: Harness Passive Vibration to Continuous Micro Finger Writing with a Commodity Smartwatch. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 1, Article 46 (March 2021), 25 pages. https://doi.org/10.1145/3448119

1 INTRODUCTION

Small-sized wearable devices like smartwatches and head-mounted devices(HMD) have grown immensely popular with today's users. However, input for these devices remains a challenging problem since they have limited or even no input space. This often leads to inconvenient interactions and limits device usage in mobile scenarios. For example, today's head-mounted devices rely on handheld controllers in one hand or both hands or require hands to be present in the field of view of a cumbersome camera. Voice input is neither applicable in a noisy place nor a quiet place, such as a night club or a library. As the capabilities of wearable devices continue expanding, so too does the demand for subtle, continuous, and always-available ways to control electronics on the go.

Authors' addresses: Wenqiang Chen University of Virginia, wc5qd@virginia.edu; Lin Chen VibInt AI Limited; Meiyi Ma University of Virginia; Farshid Salemi Parizi University of Washington; Shwetak Patel University of Washington; John Stankovic University of Virginia.

2474-9567/2021/3-ART46 $15.00
https://doi.org/10.1145/3448119

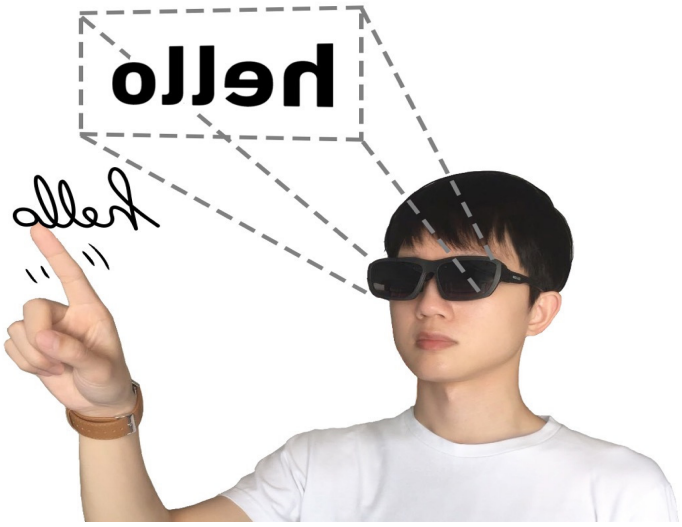

Fig. 1. A continuous micro finger writing system for AR/VR interaction

Input methods using hand or finger writing can satisfy this need. [40] Mid-air hand writing offers new opportunities for such fast, subtle, and always-available interactions, especially on devices with limited or even no input space (e.g., wearables). Similar to writing on a trackpad, using the index finger to interact with the virtual world is a natural method to perform input.

A wide variety of approaches have been considered to enable mid-air hand writing, ranging from using smart rings [18, 42, 65], to wearing gloves [33, 60], to using RF waves [6, 21, 34, 36, 50, 56]. Most common are systems that use worn cameras or depth cameras [5, 15, 20, 29, 43, 47, 55] which are generally accurate at tracking fingers spatially. However, common weaknesses across camera-based systems are the inability to detect the micro-level finger movement and the need for bulky depth sensors. Additionally, the performance of these computer vision methods is severely effected as hands move out of the field of the views (FOV) of the cameras. Therefore, most of these systems require the user to hold their hand in front of the body for interactions. Finally, being able to track micro-level finger movement is a power and computationally intensive task. Therefore, it requires a headset to have more batteries, making them heavier, and limiting these devices' ubiquity.

In this paper, we present ViFin (Vibration Finger), a continuous micro finger writing method using a commodity wrist-worn device (as shown in Figure 1). We observe that the micro finger movement causes vibrations. When users' index fingers write different numbers or letters, the finger produces small, but distinct vibrations, propagating through the hand to the smartwatch. With the Inertial measurement unit (IMU) in a commodity smartwatch we can recognize the user's finger writing in the air through finger writing vibration. Thus, ViFin can detect and recognize the vibration of continuous finger writing. Users can write numbers (0–9) and letters (a–z) with their fingers in the air continuously. The text output can be sent to any displays, such as smartglasses and smart TVs with wireless protocols.

Notably, existing IMU-based hand tacking systems can only track coarse-grained hand writing since they required the users to move their hand in a large motion so that the smartwatch on the wrist could have enough movement for distance calculation. In order to detect fine-grained continuous finger movement, existing systems required on-finger sensors, such as customized rings and gloves. It is notable that commodity devices are easily

manufactured in large volumes and have become affordable. Thus, a commodity smartwatch to recognize fine-grained finger writing is more practical and easily-accessed than are customized devices. Some existing work on gesture classification, which leverages machine learning for detecting input, only works for static gestures, but not continuous gestures. ViFin requires no cumbersome instrumentation of the hands or fingers, and works with **commodity** wrist-worn devices. Furthermore, ViFin achieves **continuous micro** finger writing. Additionally, our system works across users with relatively small training and achieves an accuracy of 90% in recognizing numbers and letters as shown through a real-time system evaluation with 15 participants. An demo video of ViFin is available at https://youtu.be/Br1LcEB84XU.

In creating ViFin, we have to overcome many challenges. First, users write in the air continuously; therefore, segmenting a single number or a letter for labeling and training is complicated. Second, different users might write the same number or letter differently. In order for ViFin to be adopted, it needs to work across different users without requiring the user to perform exhausting training. For example, some people write the number 8 in a cross stroke, while others write the number 8 with two circles. Third, ViFin should be robust to many real scenarios, such as writing during walking, with different writing speeds and gestures, and be resistant to the watch's slippage on the wrist.

In response to these challenges, we developed a preliminary study, which implements a concrete finger writing system to verify that different graphemes of finger writing have unique vibration features. Then, inspired by the Recurrent Neural Aligner [44], ViFin classifies continuous finger writing numbers and letters. Furthermore, to make ViFin work across different users, we utilize transfer learning to fine-tune different writing habits. Also, we design a calibration and update scheme inspired by active learning to further improve the accuracy. Lastly, ViFin incorporates five training free gestures for extra controls, such as "backspace", and "enter", which execute in another simultaneous thread.

A real-time system experiment shows that ViFin can work well over one month duration, even though users write graphemes with different speeds, different arm positions, and various challenging circumstances such as in a metro and on an airplane. Also, we conducted two user studies of real-world tasks.

In summary, the major contributions of this work are:

- To the best of our knowledge, we are the first to realize continuous **micro finger** level writing for a mid-air system via vibrations with a **commodity smartwatch**.
- An new algorithm is developed that uses the Recurrent Neural Aligner to classify **continuous** finger writing numbers and letters. A real-time system is implemented based on this solution.
- A comprehensive series of experiments evaluate accuracy for individuals and across users, and show the effects of different training sizes, calibration, spell checking, different writing speeds, different arm positions, various watch displacements, different environments, mobility, gesture control, and how performance varies over one month of use.

2 RELATED WORK

In this section, we present existing literature on mid-air hand writing and text entry on wearables. To summarize the following paragraphs, all existing works only recognize hand writing in large motions, or only recognize static gestures, or require customized devices, while ViFin recognizes **continuous**, **micro** finger-level writing with **commodity** devices.

2.1 Mid-air Hand Interactions

There have been some techniques proposed to facilitate fine-grained finger motion as inputs to wearables. A widely used approach consists of using a body-mounted camera to sense on-body touch input. These cameras have been placed on the head [19], shoulder [20, 47], chest [7], and wrist [24, 48]. Recognizing hand pose through depth

cameras [38, 49, 51, 52, 66] is another stream of research that investigates capturing the detailed hand posture. Cameras have a field of view (FOV) problem and require high calculation resources. WiFi-based [6, 21, 34, 36, 50] and acoustic-based [39, 57] methods can also achieve micro finger writing. However, these systems also rely on line-of-sight between the source and the sensors. And the obstacles surrounding the transmitter and receiver influence reflections.

There has also been work in using bioimpedance [70], capacitive sensing [54, 61] and ultrasound [26] for discrete hand gesture classification from a wristband. These methods result in discrete, gesture-based interactions, while ViFin recognizes **continuous** text writing. Kim et al. used infrared illumination from a customized wrist to track hand pose [29]. Project Soli [56] uses radio frequency signals to recognize subtle finger gestures. Pyro enables micro thumb-tip gesture recognition with a pyroelectric passive infrared sensor [17]. Smart rings provide an attractive form-factor for always-available subtle input. Magic Finger [64] and Light Ring [28] enables users to interact with surfaces using optical sensors embedded in a wearable ring form factor. Researchers have also used magnetic tracking for wrist and finger tracking [8, 9, 42]. Another approach is the use of gloves or finger strips to estimate finger and hand pose [23, 25, 32, 40, 59, 60, 68, 69]. However, all these mid-air finger writing systems require customized devices attached to the finger, while ViFin achieves finger writing with a **commodity** smartwatch, which is more practical and easily-accessed than customized devices.

2.2 IMU Gesture Tracking and Recognition

Due to their rich sensing capability and small size, IMUs have been used to sense various human activities. Based on the IMU displacement calculation, it is capable of tracking the hand with the IMU on the wrist. Armtrack [45] tracked the users' arm/hand with a smartwatch by calculating the accumulated displacement of the smartwatch. Zhou et al. [71] utilize a kinematic chain to track human upper limb motion by placing multiple devices on the arm. Cutti et al. [16] utilize the joint angles to track the upper limbs' movements by placing sensors on the chest, shoulder, arm, and wrist. AirContour [67] obtained gesture contour and recognized it as characters. SignSpeaker [22] used IMU to recognize sign language with hand gestures. These works all required large motions to move the arms/sensors, which may get uninvited attention and be perceived as strange by people.

On the other hand, many works use machine learning based approaches to classify hand gestures. RisQ [41] detected smoking gestures and sessions with a wristband and used a machine learning pipeline to process sensor data. Thomaz et al. [53] described the implementation and evaluation of an approach to infer eating moments using a 3-axis accelerometer in a smartwatch. Xu et al. [63] build a classifier to identify users' hand and finger gestures utilizing the essential features of accelerometer and gyroscope data measured from a smartwatch. Viband [31] leveraged accelerometers of a smartwatch to classify hand gestures, such as flicks, claps, scratches, taps, and can also recognize grasped motor-powered objects. Serendipity [58] leveraged accelerometer and gyroscope to recognize five statics discrete gestures, such as pinch, tap, rub, squeeze and wave. Shi et al. used a finger mounted IMU to detect gesture on a surface [46]. WRIST [65] explored combining IMU data from a smartwatch and smart ring for distal pointing and gesture interaction. Chen et al. [10–14, 62] utilized passive vibration to recognize on-body tapping gestures. However, all these machine learning based systems only work for statics and discrete gesture recognition, which is not qualified for continuous text writing.

To summarize, all these IMU systems either calculate the IMU displacements to track the hand movement from a large motion, or classify the static, discrete gestures based on machine learning approaches. However, ViFin recognizes **continuous** and **micro** finger writing, which is more natural and practical than static gestures classifications or large motion tracking.

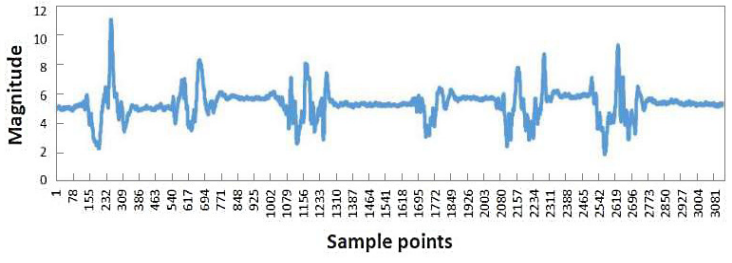

Fig. 2. Discrete grapheme of 1, 2, 3, c, a, r.

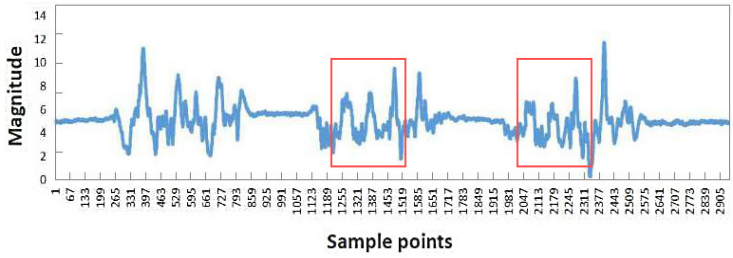

Fig. 3. Continuous grapheme "123", "car", "cat". The red box shows the front part of the vibrations of the words "car" and "cat".

3 PRELIMINARY STUDY

3.1 Observation

When people pronounce different words, their throats vibrate differently. These different vibrations propagate through the air from one person's throat to another person's ears and that's the way they communicate. With microphone sensors and a speech recognition technique, machines can also understand human speech by detecting the unique sounds in the air. We observe that when people write different graphemes in the air with their fingers, this produces different vibrations too. Although micro finger writing movement is too week to make detectable sounds and propagate in the air, it does propagate through the hand to the wrist. This vibration can be detected by an inertial measurement unit (IMU) sensor in a commodity smartwatch worn on the wrist. Figure 2 and Figure 3 show the writing vibration signals of z-axis of accelerator in the smartwatch. As shown in Figure 2, a person wearing a smartwatch writes single discrete grapheme of 1, 2, 3, c, a, r. It shows that different finger writing graphemes have different vibration signals. However, as humans talk continuously, people also write numbers or words by continuous finger writing. Thus, we ask the person to write continuous numbers "123". We also ask them to write letters continuously with two words, "car" and " cat", as shown in Figure 3. We observe that these three vibration signals are different. Also, the front part of the vibrations remains similar between "car" and " cat" in the red box as shown in Figure 3. Although we can not find the specific point of transition between each number in the continuous vibration of "123", we observe that discrete numbers 1,2,3 and continuous numbers "123" share some common features as shown in Figure 2 and Figure 3. Thus, we think these different micro finger vibrations may be detected and recognized as continuous text input.

In the following subsections, we verify that each finger writing vibration for a specific number or letter has unique features and can be classified. Thus, we collect finger writing vibrations for 10 numbers (0–9) and 26

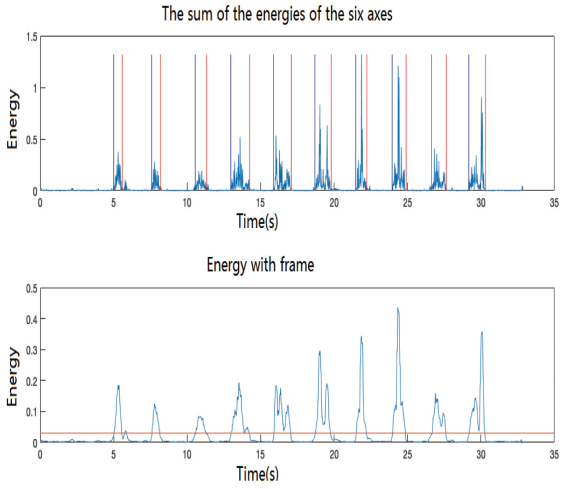

Fig. 4. Combining data from six axes for segmentation based on energy

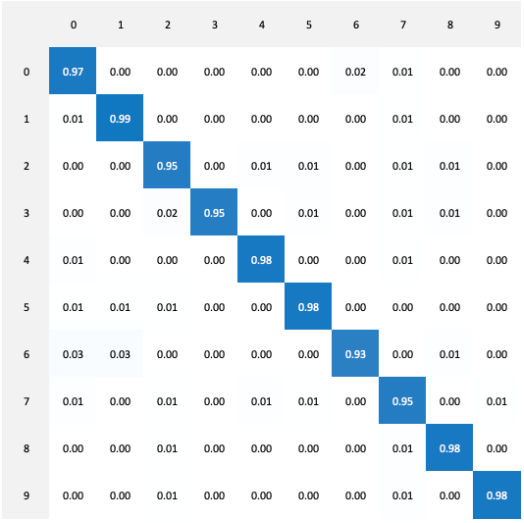

Fig. 5. Confusion matrix of numbers

letters (a–z). When a user writes discretely, there is a short pause between graphemes. We detect each pause (much lower magnitude) to segment each number/letter, then label each of them manually. In addition, a simple neural network (NN) is utilized to classify the different numbers and letters.

3.2 Data Collection

The user is asked to wear a commodity smartwatch (Huawei Watch 2) on her wrist, then write numbers and characters in the air with her index fingers. The IMU sensor in the smartwatch detects the smartwatch vibration induced by finger writing. To further investigate our initial hypothesis, we asked a single user to write the numerical numbers from "0-9" and the 26 alphabetic letters from "a-z" thirty times. During writing, a one-second pause is required between each number or letter in order to manually segment and label the data. The user in this study is 27 years old with 21.3 body mass index (BMI). Note that the experiments were approved by the relevant institutes in our university.

3.3 Segmentation

After collecting the vibration signal, we segment the signals on the smartwatch. One current challenge is that the IMU signals are too weak to be segmented, because the smartwatch is worn on the wrist instead of the index finger directly. With ViFin, we combine 6-axes signals to improve the SNR (Signal Noise Ratio) to segment different finger writing samples more easily. Next, our system uses an energy-based threshold approach detailed in [35] to detect the starting point of the finger writing signals. In this example, we set the frame length as 0.2 seconds, the frame-shift as 0.01 seconds, and the energy threshold as 0.03. Figure 4 shows the results of the segmentation for the numbers 0–9.

3.4 Aligning

Signal starting points detected by the threshold approach will inevitably have a certain deviation. The deviation between different signals will increase the classification difficulty, especially when we use time-domain

Table 1. Accuracy of discrete graphemes in different environments

Quiet office room (44 dB)	Noise office room (85 dB)	Subway (65 dB)	Airplane (77 dB)
96.8%	96.9%	97.2%	95.4%

information as features. If we try to calculate the distances between signals without aligning them first, we will get inaccuracy with large distance results; this occurs even when the distance calculated is the same signal but misaligned. In order to solve this problem, it is necessary to align the signals before classifying them. In this work, we utilize the GCC (general cross-correlation) based algorithm [30], which is shown in Equation 1 to align signals.

$$r_{xy}(m) = \sum_{n=-\infty}^{\infty} y(n)x(n-m) \tag{1}$$

Where $r_{xy}(m)$ is the cross-correlation function, x and y are two signals from the same class, and m is the parameter of time shift.

3.5 Classification

We use an artificial neural network (ANN) based on a gradient descent method that minimizes the sum of the squared errors between the actual and the desired output values. Before inputting the aligned vibration signals into the classifier, the Z-score normalization [4] is used to reduce the variation of the signal. Note that we set the number of hidden layers to 2 and the number of hidden nodes to 20.

3.6 Results

We used three-fold cross-validation to evaluate the accuracy. The accuracy of numbers and letters is 96.9% and 97.1%, respectively as Figure 5 and Figure 14 show. We also observed that the average accuracy is 97% with aligned signals using GCC, while the accuracy is 93.5% using dynamic time warping (DTW) instead of GCC. With neither GCC nor DTW, the accuracy is 90.9%. Thus, we proved that different numbers and letters of finger writing have unique vibration profiles. However, to make ViFin practical for real applications, we further explore two questions. First, real users not only write discretely, but also in a continuous fashion. How can we segment and label a single grapheme in continuous writing? Second, different people write the same grapheme in different ways. How can we make a finger writing system work across different users?

3.7 Resistance to Noise

To see how the environment affects finger writing, we did the same experiments, but in four different scenarios: a quiet office environment (i.e., the control group), a noisy office with loud music, a subway, and flying on an airplane. Note that we finished this experiment before COVID 19. These four environments are the typical representation of our daily lives (ranging from casual noise to loud noise) in which a finger writing system could be used. Our test scenarios represent a variety of noise levels, ranging from 44 to 85 dB. Table 1 enumerates the average accuracy in different environments. The results demonstrate remarkably good performance comparing to the control group in a quiet office (96.8%). It shows that the finger writing vibrations keep unique profiles in different environments and are resistant to the surrounding noise.

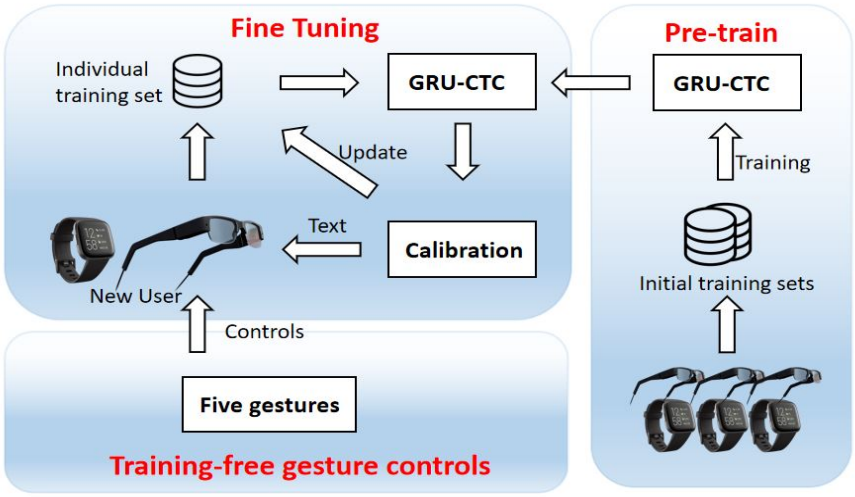

Fig. 6. Overview: A new user initializes ViFin with a small training set. Then ViFin fine-tunes the pre-trained model, which is trained with a huge previous training data, to recognize finger writing. Five training free gestures are also recognized to support extra controls.

4 VIFIN SOLUTION

4.1 Overview

ViFin is a real-time standalone micro finger level writing recognition system using passive vibrations, which are detected by a commodity smartwatch. First, it allows users to write numbers and letters in both discrete and continuous fashion using a GRU-CTC based model. However, it has two challenges. One is that it requires exhausting initial training. Another one is that it requires every user to collect training sets because different users write graphemes in a different fashion. In order to reduce the training time and effort required to initialize the system for a new user, ViFin utilizes pre-training and fine-tuning on new users. Besides, we design a calibration and update scheme to improve accuracy further. Unfortunately, a real-time text input system that only recognizes numbers and letters does not work because it requires extra controls, such as switching input methods between a number and a letter, backspace, enter, etc. Therefore, ViFin incorporates five training free gestures for extra control, which execute in another simultaneous thread. Last, the results are sent to the smartglasses by using the Bluetooth protocol.

As Figure 6 shows, a group of people collect a large amount of initial training sets for training a pre-trained GRU-CTC model. For a new user, a small individual training set for fine-tuning the GRU-CTC model is collected. To further improve the accuracy, ViFin has a calibration and update scheme. In another thread, ViFin recognizes five gestures for the extra controls, such as "backspace", "enter", etc. Details for each of the main parts of our solution are presented below.

4.2 Continuous Writing

When users write discretely, we can detect the pauses between graphemes. Thus, we can use a one-input-one-output NN model to classify different numbers and letters. However, what should we do if there is no pause

between numbers or letters when users write continuously? Inspired by speech recognition techniques, we can use long short term memory networks to recognize an entire continuous signal of speech by a sequence to sequence model. Thus, for continuous writing we can use a sliding window to recognize each time-step of a vibration with memory networks.

However, there are two challenges to use sliding windows. For example, we can use 4 windows to cover the continuous finger writing vibration signal of the letter "a" and the letter "b." First, we don't know where the transition point is between the letter "a" and the letter "b". We can label the 4 windows as [a, a, b, b] or [a, b, b, b] or [a, a, a, b]. To overcome this challenge, we can use the method of exhaustion by going through all the cases. To be specific, we do not require an alignment between the vibration signals(input sequence X) and the labels (output sequence Y). To get the probability of an output Y given an input X, we sum over the probability of all possible alignments between the two. To be precise, for a single (X, Y) pair the conditional probability is shown in Equation 2 where $p_t(a_t|X)$ is the probability for a single alignment step-by-step and $\sum$ marginalizes over the set of valid alignments.

$$p(Y|X) = \sum \prod_{t=1}^{T} p_t(a_t|X) \tag{2}$$

Second, the method of exhaustion is time consuming. To make it more efficient, we can do dynamic programming using the Forward-backward algorithm in the Hidden Markov Model (HMM) [1] to get results of classification for 4 windows, such as [a, a, b, b]. With a naive trimming, we can reduce the repeated letters and get the correct results of [a,b]. However, users may write the same letters repeatedly. For example, there are two repeated "l" in the word "hello." Inspired by the Recurrent Neural Aligner (RNA) [44], we add another class, labeled "∅". RNA is a neural network with memory. Thus, if the result of the first window is "a", the result of the second window will be forced to be "∅" or "b" even if it may be more familiar with the writing vibration of "a". Thus, the correct result of the four windows can be $[a, \varnothing, b, \varnothing]$. For the word of "hello", the correct results can be $[h, e, \varnothing, \varnothing, l, \varnothing, l, \varnothing, \varnothing, o, \varnothing]$ if there are 11 windows covering the entire word.

4.2.1 Neural Network Architecture. We use unidirectional Gated Recurrent Units (GRU) with an extra connectionist temporal classification (CTC) layer [44] as shown in Figure 7. GRU is an efficient Recurrent Neural Network that has memories, but has fewer gates (reset and update gates) than Long Short-Term Memory (LSTM). CTC aligns the input and output with a token "∅" and computes the loss much faster using the Forward-backward algorithm than the method of exhaustion. To be specific, the GRU has one hidden layer and 32 neurons. We choose Adam to optimize network training. The sliding window size (time frame) is 0.5 seconds and the moving step is 0.1 seconds. Since IMU has six axes of data, we concatenated each window's data of six axes in series as a single input to the GRU. We also utilize the Z-score normalization (see the section 3.5) to reduce the variation of the signals before they are put into the GRU neural network. We use a full connected layer after GRU, following a softmax layer and CTC loss.

4.2.2 The Training Set and Augmentation. To remember every transition between graphemes, we collect each pair of numbers (00, 01, ..., 98, 99) and each pair of letters (aa, ab, ac, ..., az, ..., za, zb, zc,..., zz) as the training set. We require participants to pause for a second between pairs of graphemes, then we segment each pair and label it based on energy threshold (see section 3.3). In total, there are 100 pairs (10*10) of numbers and 676 pairs (26*26) of letters of training data which can cover all transitions between every two numbers and letters. However, RNN training requires large training data. Thus, we further achieve data augmentation. To be specific, We combine pairs of consecutive numbers or letters into a set of 4 graphemes as additional training samples. We add (00-01, 01-02, 02-03, ..., 98-99) into training set for numbers and (aa-ab, ab-ac, ac-ad, ..., zy-zz) for the letters, respectively.

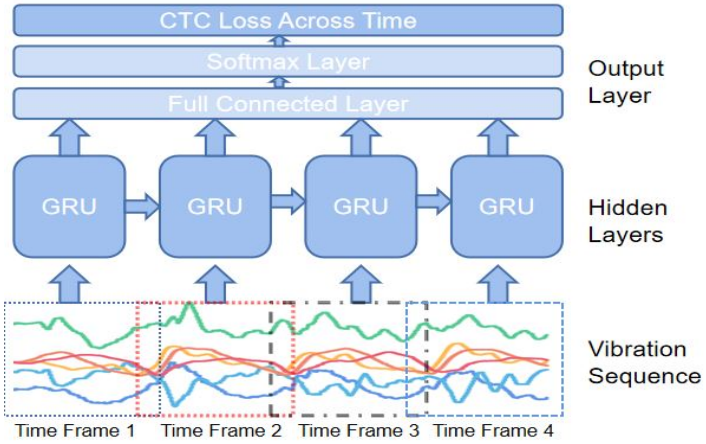

Fig. 7. GRU_CTC architecture.

Note that "−" is the pause (background noise) between two pairs of graphemes, which also provide information about the finger state of rest for GRU_ CTC neural networks.

4.3 Across Different Users

One of the great challenges of the ViFin system is the ability to work between different users without exhausting training and calibration as not only people have different writing practices, but also they have different wrist sizes and shapes and different hand anatomy. To highlight this, take the number "1" as an example. A person can write this as a straight vertical line while a different user may write a variant that resembles more a font-like number, i.e., having conventional extremities rather than simply a straight vertical line. Requiring a new user to initialize ViFin with exhausting training is impractical. The question is how to reduce the training time and effort required to initialize for a new user. Fortunately, graphemes written by different people share common features. Thus, it is possible to reduce the training time and effort required to initialize for a new user by learning something from other users.

Inspired by transfer learning, ViFin generalizes the knowledge that has been learned in previous data collected from other people to a new person. First, we use the aforementioned neural network architecture (GRU_CTC) to generate a pre-trained model according to a large training data set built through an initial user study. Then, we fine-tune the model with a small training data from the new user. To be specific, We first freeze the weight of the top-layers of the model in order to preserve the knowledge learned by the previous process. Then, we train the model with new data with a small learning rate. In this step, the model will slowly adapt to the new person's input and adjust to the covariate shift between previous data and new data. The convergence of this step is quick, making fine-tuning step much faster than training from the scratch (see section 5.5).

4.4 Calibration and Update

In order to further improve the accuracy and user experience of ViFin, we add a spell check. [2] Note that an alternative is to use a language model. The spell check function tries to choose the most likely spelling correction for words. There is no way to know for sure (for example, should "ca" be corrected to "car" or "cat" or "cia"), which suggests we use probabilities. We are trying to find the correction, defined as c, out of all possible candidate

corrections, that maximizes the probability that c is the intended correction, given the original word w:

$$argmax_{c \in candidates} P(c|w) \tag{3}$$

By Bayes' Theorem this is equivalent to:

$$argmax_{c \in candidates} P(c)P(w|c)/P(w) \tag{4}$$

where P(c) is the probability that c appears as a word of English text in a specific or general application. Since P(w) is the same for every possible candidate c, we can factor it out, giving:

$$argmax_{c \in candidates} P(c)P(w|c) \tag{5}$$

Based on the spell check, ViFin comes up with a small number of specific graphemes that are usually recognized wrong. These errors might have happened because the fine-tuning model is not tuned perfectly with a small training set, or users might not write these graphemes well. However, requiring users to write another entire round of training set again is time-consuming. To compensate for this limitation, we design a more efficient update scheme with part of the entire training set. Inspired by active learning, ViFin suggests that users collect a few pairs of graphemes of training data for graphemes that are error prone. Also, users can choose the graphemes with poor accuracy by themselves to update the training set, especially for numbers which can not be corrected by spell check. For example, when users find out number 1 is usually misidentified, then they can choose number 1 in the update schemes and start to collect number pairs which contain the number 1. Then this new training data and the previous training set will be input into the fine-tuning network and produce an updated model.

4.5 Training-free Gesture Control

For a real-time standalone text input system, it is essential to have some other operations beyond writing numbers and letters as input, such as backspace, enter, etc. Also, we need operations to wake-up ViFin and switch the input method between numbers and letters. We seek to have these gestural inputs work across users without any training. We leverage the gyroscope in the smartwatch for detection of these gestures.

4.5.1 Gyroscope Angles. The gyroscope in a smartwatch detects the rotation angles of the smartwatch for three axes of x, y, and z, which decide the rotation angles of the smartwatch according to the relationship between smartwatch coordinate system and geographic coordinate system. Inspired by the language of aeronautics, we define the x-axis rotation of the smartwatch as the pitch angle, which changes when the hand is raised or lowered based on the elbow. The pitch angle increases when a user raises their hand. Y-axis rotation of the smartwatch is the yaw angle, which changes when the hand move left or right based on the elbow. Moving left increases the yaw angle. Z-axis rotation is the roll angle, which changes when the wrist rotates. The roll angle becomes larger when the wrist rotates to left. Although gyroscope produces rotation angular velocity, we calculate the value of the angle by the following equations:

$$\theta(t) = \sum_{i=1}^{t} \frac{w(i)}{f} * \frac{180^\circ}{\pi} \tag{6}$$

Where $w(i)$ is the rotation angular velocity at time t, $\theta(t)$ is the value of angle at time t, and f is the sampling rate.

Based on the gyroscope angle variation, we can easily detect the hand movements. For example, Figure 8 (a) shows that the value of the pitch angle goes up then down when the hand is raised then back, while other angles only change mildly. Figure 8 (b) shows that the value of the pitch angle goes down then back when the hand goes down then back, while other angles only change mildly. Thus, Firgure 8 shows that by detecting which axis has a significant half wave, we can map it to recognize different hand movements.

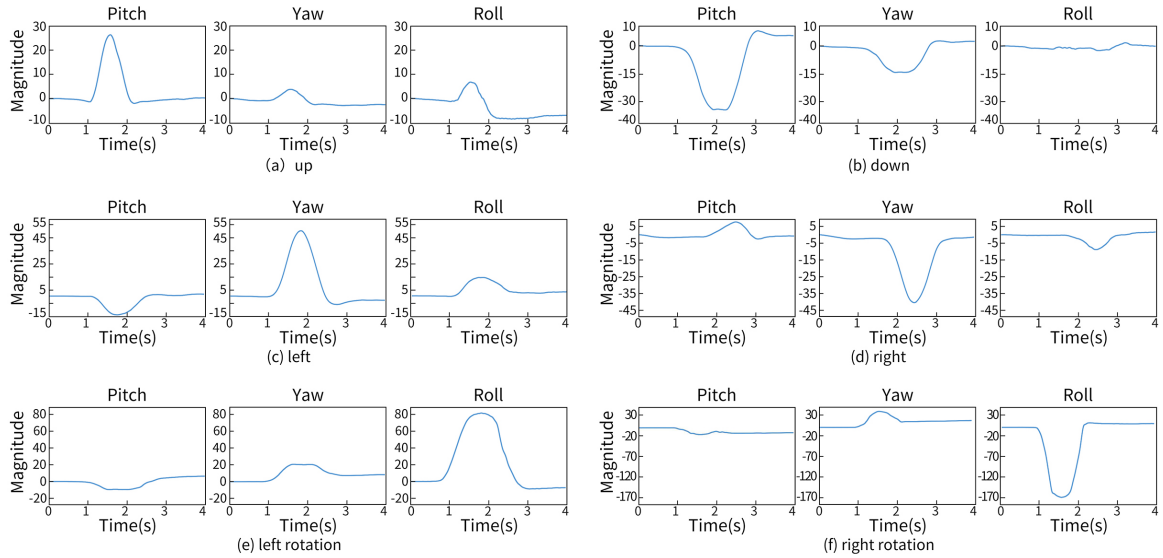

Fig. 8. ViFin detects which axis of the smartwatch's gyroscope has a significant half wave, then maps it to recognize different hand movements.

To be specific, we first find out the axis which has the largest variation. Then, we determine whether there is a peak or a valley. If yes, we further determine whether the value of the angle is back to quiet (smaller than the 40% of the absolute value of the peak or the valley). If still yes, we recognize the signal as one of the hand movements. This training free algorithm runs in another simultaneous thread with the finger writing algorithm. When both algorithms detect a vibration event (writing or gestures) and get results, we only output a training free gesture, so that the gestures are not interpreted as finger writings. Also finger writing does not involve a large distinct pattern of angle variation as shown in Figure 8. Only gestures show these patterns, so that the finger writings are not interpreted as gestures.

Since it is difficult to rotate the hand to the left, we do not recognize this gesture. Thus, ViFin detects five training free gestures, including rotation (the wrist rotates to the right), left (moving the hand left then back), right (moving the hand right then back), up (moving the hand up then back), and down (moving the hand down then back). These training free gestures can be assigned to different controls.

5 EVALUATION

The evaluation mainly includes two parts. One is a baseline accuracy with off-line collected data, which studied the detection rate of graphemes, recognition accuracy of letters and numbers, the accuracy across different users, the accuracy with different training sizes and the accuracy with a calibration scheme. For the other one, we further studied the accuracy of a real-time system under different conditions, including different writing speeds, different arm positions, different watch location displacements, different environments. In this real-time evaluation, we also evaluated the performance when users wrote during walking. In addition, we conducted an experiment over one month, which showed temporal stability.

5.1 Implementation

We have implemented ViFin as a standalone application program on multiple commodity Android smartwatch, including the Huawei Watch 2, ASUS ZenWatch 2, and Jeep watch. ViFin utilizes the built-in accelerometer and gyroscope and acquires the motion readings through existing Android Wear APIs to detect the finger writing vibration. The maximum sampling rate through the APIs is 100 Hz, 200 Hz, 400 Hz for the Huawei Watch 2, ASUS ZenWatch 2, and Jeep watch, respectively. Through the short time Fourier frequency spectral analysis, we observe that the frequency features of vibration signals caused by finger writing should be below 50Hz. Thus, we down-sampled all of the watches into a 100 Hz sampling rate. We have implemented ViFin on commodity smartwatch to recognize the micro finger writing in real-time. Finally, the results of finger writing are sent from smartwatch to Madgaze smartglasses via Bluetooth communication with Service Discovery Protocol(SDP) and RFCOMM Socket.

For initial data training, we developed an app, name VFC (ViFin Collection), on laptops and smartglasses. VFC shows each pair of graphemes in order. When VFC detected a finger writing signal, it shows another pair of graphemes. We did not have enough smartglasses for all participants so we used the screen of laptops instead. To connect smartwatch and laptops, we made both of the devices connect to the same WIFI, then used network socket [3] to send collected data from the smartwatch to the laptop. After that, we trained neural network models with Tensorflow on the laptop and sent the trained models back to the smartwatch. The laptop we used has Intel Core i7 CPU (1.8 GHz) and 16 GB RAM. And Huawei watch 2 has a 1.2 GHz Quad-Core processor and a RAM with 512 MB.

5.2 Data Collection

5.2.1 Participants and Protocol. Due to the quarantine caused by COVID-19, we recruited 4 supervisors to collect data from their families. We guided them on how to collect data virtually in Zoom. 15 participants (9 males and 6 females) were recruited for this study. One of them was left-handed participant. They were in the age range from 11 to 58. Additionally, their body mass indexes (BMIs) range from 17.13 (lean) to 30.38 (obese). They were required to have a seat and place their elbow on the table during writing to avoid extra hand movement noise for the training data labelling. We played a two minutes video of how to collect data, then they were required to have a 10-minute warm up. In total, 58200 (776 pairs $\times$ 5 times $\times$ 15 participants) pairs of training graphemes were collected. Note that all the experiments involving human subjects conformed to the relevant regulations of our institute.

5.2.2 Training Set. In order to learn every transition between graphemes, we collected each pair of numbers (00, 01, ..., 98, 99) and each pair of letters (aa, ab, ac, ...za, zb, zc,..., zz) for training, which had 100 pairs of numbers and 676 pairs of letters. Participants were required to repeat the training set collection five times (rounds). In total, there were 3880 pairs (776 $\times$ 5) of graphemes in the training set for each person before data augmentation (see section 4.2.2).

5.2.3 Test Set. We collected additional data for testing ViFin. To evaluate how good ViFin can detect numbers, 100 random numbers were collected for each person. These 100 numbers were split into 10 groups and participant were required to write them continuously. The random 100 numbers were grouped as follow: [1,3,2,4,6,8,8,9,1,3]- [2,0,5,4,7,4,8,5,9,7]- [9,9,7,1,4,0,5,3,3,0]- [7,1,8,1,3,3,3,0,2,5]- [8,7,9,6,9,5,2,9,7,4]- [8,9,5,8,7,5,5,2,3,6]- [2,1,6,1,6,8,5,2,0,6]- [2,4,8,4,4,0,6,7,2,0]- [6,0,4,4,5,6,3,1,7,9]- [3,6,2,1,7,9,0,8,1,0]. Note that some numbers were repeated in the same line. In order to balance each letter for evaluating the accuracy of each letter, we collected seven English sentences, where each sentence contain 26 different letters. The first sentence was 26 letters in order from a to z. Participants were required to write them continuously. The rest of the six sentences are called perfect pangrams and consisted of some English words. In perfect pangrams, each letter of the alphabet occurs once and only once. Between

different words, participants are required to pause according to their writing habits. VFC showed each sentence in order. The following letters are the ground truth of seven sentences, which contain each letter seven times equably: [abcdefghijklmnopqrstuvwxyz]- [mr jock tv quiz phd bags few lynx]- [cwm fjord bank glyphs vext quiz]- [blowzy night frumps vexd jack q]- [squdgy fez blank jimp crwth vox]- [tv quiz drag nymphs blew jfk cox]- [q kelt vug dwarf combs jynx phiz].

5.3 Evaluation Metrics

For each sentence of numbers or letters, we calculated the similarity between ground truth and the recognition results by minimum edit distance, which counts the minimum number of operations required to transform the recognition results into the ground truth. One of the simplest sets of edit operations is defined by Levenshtein in 1966 [27], which allows deletion, insertion, and substitution. Using Levenshtein's original operations, the edit distance from $a = a_1...a_n$ to $b = b_1...b_n$ is given by d_{nm}, defined by the recurrence [27].

$$
\begin{aligned}
&d_{i0} = \textstyle\sum_{k=1}^{i} W_{del}(a_k), && \text{for } 1 \le i \le m \\
&d_{0j} = \textstyle\sum_{k=1}^{j} W_{ins}(b_k) && \text{for } 1 \le j \le n
\end{aligned}
$$

$$
d_{ij} = \begin{cases} d_{i-1,j-q} & \text{for } a_i = b_j \\ \min \begin{cases} d_{i-1,j} + W_{del}(a_i), \\ d_{i,j-1} + W_{ins}(b_j), \\ d_{i-1,j-1} + W_{sub}(a_i, b_j), \end{cases} & \text{for } a_i \neq b_j \end{cases} \quad \text{for } 1 \le i \le m, 1 \le j \le n \tag{7}
$$

$W()$ in equation 7 is the cost function to count the weight number of operations, where we set weight as one.

We defined the recognition accuracy as $1 - (edit\_operations/text\_length)$. According to the minimum edit distance, we also calculated the detection rate by detecting which graphemes are inserted. Also, we obtained the accuracy of each number or letter by detecting which graphemes are inserted and substituted.

5.4 Baseline Accuracy

In this subsection, we study the detection rate and accuracy of ViFin. In addition, we examine how the accuracy is affected when ViFin works across different users, with varying sizes of training data, and with/without the calibration and update scheme.

5.4.1 Detection Rate. For a real-time system, the recognition results are supposed to be immediately shown when users start to write. It is important to determine how sensitive the system is to detect the users' writing and display the results. The detection rate shows how sensitive the system is in detecting the finger writing. According to the minimum edit distance, the detection rate for numbers and letters was 96.5% and 97.1%, respectively.

5.4.2 Recognition Accuracy. We first evaluate the recognition accuracy for continuous number and letter writing by using a GRU_CTC model. We used each person's five-round training set to train individual GRU_CTC models for each person. The average accuracy of 15 participants was 89.7% and 90.6% for numbers and letters with a standard deviation of 3.2 and 3.6, respectively. The average accuracy of graphemes was 90.2%. Without data augmentation, the average accuracy was only 78%, which shows that data augmentation contributes significantly to the system. Figure 9 shows the average accuracy of each number and letter. We noticed that the letter "a" had the lowest accuracy at 0.62%, which was mistakenly recognized as letter "q" most of the time, while the letter "q" had an accuracy at 100%.

5.4.3 Across Different Users. Although the recognition accuracy is 90.2% with five rounds of the training set for one person, it is difficult to ask a real user to collect five rounds (5 hours) of initial training data before using ViFin. Thus, we used 14 out of 15 participants' training data to train a general model then test on the data not

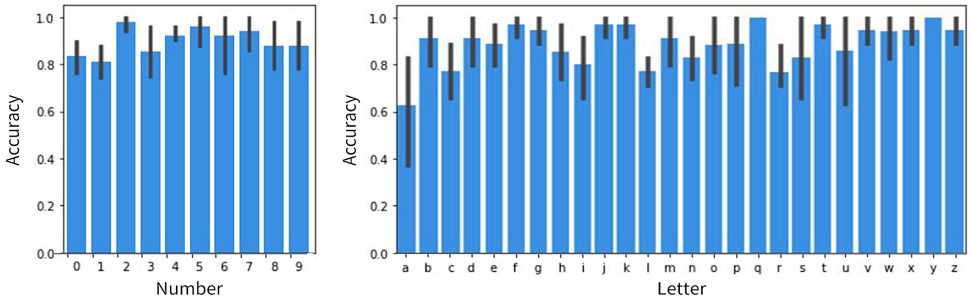

Fig. 9. The average accuracy of numbers and letters.

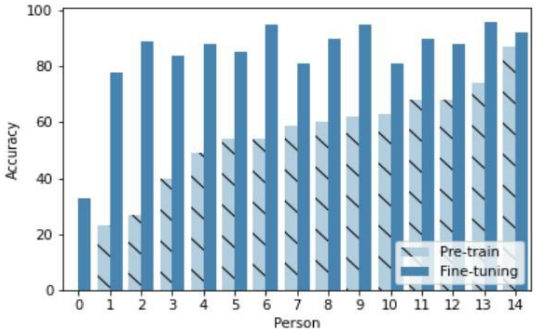

Fig. 10. The accuracy of pre-trained models and Fine-tuned models for 15 participants.

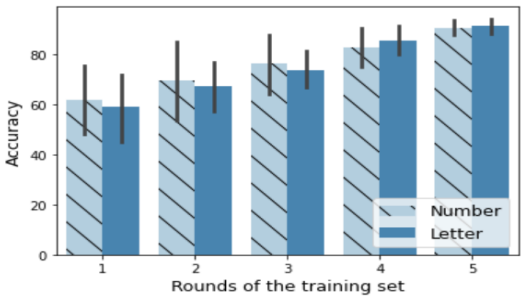

Fig. 11. The accuracy of individual model increase with more rounds of the training set.

included in training. In this way, we repeated the training 15 times, using a different data set each time as the test set. Figure 10 shows the results. The average accuracy is 52% . We believe that this is because different people write the same graphemes in different fashions. Note that participant 0 has an accuracy of 0%. He is the only left-handed participant; therefore he writes totally different from the others. Then, we used the general model from 14 people as a pre-trained model and fine-tuned with 1 round of data from the test user. The results are also shown in Figure 10. The average accuracy of the fine-tuned model is 86%, which is much higher than the 52% from the pre-trained model. However, the accuracy of fine-tuned model of the left-handed participant is still only 33%. We believe that his fine-tuned model did not get much useful information from the pre-trained model. If we do not include this extremely low accuracy of the left-handed participant, the average accuracy of fine-tuning is 90%, which is very close to the accuracy of the individual model with 5 rounds of training data (90.2%). Thus, the fine-tuned model reduces 80 percent of the training data, but stays at the same accuracy. One implication of these tests is that we need to train separate solutions for right and left handed people.

5.4.4 Different Training Sizes. To analyze how the training size will affect the recognition accuracy, we trained models with different training sizes then obtained the recognition accuracy for each individual. Except for a training model with 5 rounds of the training set, which includes 3880 pairs of graphemes (776 pairs × 5 rounds) as in the last paragraph, we also trained models with 4, 3, 2 , and only 1 rounds of the training set, separately. Figure 11 shows that the recognition accuracy decreases of individual models (GRU_CTC) when we get smaller training sizes.

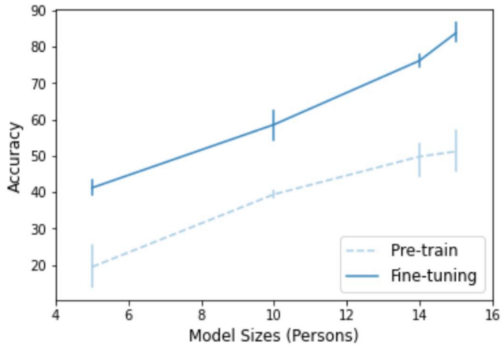

Fig. 12. The accuracy of models with different number of persons' training data for pre-trained models.

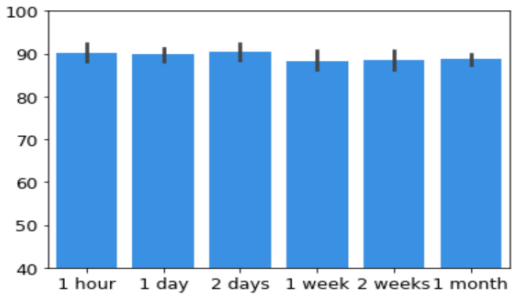

Fig. 13. The accuracy of ViFin keeps consistent during one month.

We also investigated how the different training sizes will affect the performance of the pre-trained models and fine-tuned models. We randomly chose some numbers of user's five-round data for a pre-trained model, then test on another user's test set with a fine-tuned model or without a fine-tuned model, where the number is 15, 14, 10, and 5 random users data separately. We repeat these experiments five times. As shown in figure 12, the average accuracy of pre-trained models decreases from 51.2% to 19.4%, while the average accuracy of fine-tuned models decreases from 84% to 41.2%. This shows that as the number of users increases (more training data for pre-trained model), the performance of ViFin will rise.

5.4.5 Calibration and Update. As mentioned before, the accuracy without a spell check was 89.7% and 90.6% for numbers and letters, respectively. We used word-level spell checking here, which was based on Bayesian probability. It improved the average accuracy from 90% to 98%. Since we could not use spell check for the test set of numbers, we decided to calculate each number and letter's accuracy based on the minimum edit operations, then chose two numbers and two letters with the poorest accuracy to update the training sets. Then we updated the fine-tuning model with a previous one-round training set and the new training data from another round of the training set. The updated results without a spell check showed that the average accuracy of numbers improved from 89.7% to 95%, while the average accuracy of letters improved from 90.6% to 96%.

5.5 Real-time Evaluation

In this section, we completed a real-time finger writing system, then tested its accuracy in practice. We tested the accuracy of five training free gestures. We also wanted to know the accuracy when users write graphemes with different speeds and different arm positions. How does it affect the accuracy if the smartwatch slips on the wrist or is used in different environments, such as an office or a metro? Can users write in the air during walking? How will the accuracy change over one month? Lastly, what is the time and energy consumption of the ViFin system?

We recruited 5 participants (three males and two females) out of the previous 15 participants, in the age range from 19 to 26. Their body mass indexes (BMIs) range from 17.16 (lean) to 29.28 (obese). To test the robustness of ViFin, participants were asked to write the same test sets of 100 random numbers and seven English sentences in Section 5.2.3, but write them multiple times in different conditions mentioned above. Before the test, we used each participant's first-round training data to fine-tune the model with a pre-trained model, which contains the previous other 14 participants' training data with 54320 pairs of graphemes.

5.5.1 Gesture Control. To evaluate the accuracy of five training free gestures, we required participants to perform these five gestures 20 times. The result is quite positive: each gesture has average accuracy at 100%. These gestures

Table 2. Accuracy in tough circumstances

Quiet office room	Noise office room	Subway	Walking
90.4%	90.1%	90.2%	89.5%

would be further evaluated in the user study section, where users need to perform them as ‘"wake up", "switch input methods", "backspace" and "enter" to search movies in smartglasses. (see section 6)

5.5.2 Different Speeds. Participants were required to write the test set (100 random numbers and 7 English sentences) two times with a slow speed and a faster speed. We told the participants that the slow speed means they need to write a little slower than they normally do, and fast speed means they need to write a little faster than they writes normally. Then, we allowed the users to decide the low speed and high speed. The accuracy for numbers and letters with slow speed is 90% and 91%, respectively. The accuracy for numbers and letters with a faster speed is 89% and 90%, respectively. This shows that ViFin accommodates different writing speeds.

5.5.3 Different Arm Positions. In practice, users might have different arm positions when they are writing in the air. To evaluate the impact of such variations, we tested ViFin under five different pose of wrist : (1) Pose 0 indicates the plane of the hand is parallel to the ground, (2) Pose 1 indicates that the arm rotates 45 degrees left from pose 0, (3) Pose 2 indicates that the arm rotates 45 degrees right from pose 0. (4) Pose 3 indicates that the arm rotates 45 degrees upward from pose 0. (5) Pose 4 indicates the arm rotates 45 degrees downward from pose 0. The participants were required to write the test set with the above 5 different arm positions. The average accuracy of arm positions 0-4 is 90%, 92%, 91%, 90%, 89%, respectively. This shows that ViFin works with different arm positions.

5.5.4 Watch Displacements. We further measured the smartwatch displacement, which might impact the reliability of ViFin. We asked participants to write a test set with 5 different smartwatch locations each. Location 0 means participants wear the smartwatch on the wrist closest to the fingers, with a comfortable tightness. We asked participants to write 5 rounds of the test set with 5 smartwatch locations each, which moved the smartwatch away from the fingers by 1 mm, 5 mm, 10 mm, 20 mm, and 30 mm. We observe that users can not move the watch more than 30 mm away from location 0 because the arm becomes thicker. The accuracy for 5 locations is 90%, 89%, 88%, 91%, 89% respectively, which shows that ViFin accommodates different watch displacement on the wrist.

5.5.5 Mobility. To investigate how mobility affects the performance of ViFin, we asked participants to write and walk simultaneously. The result shows the robustness of ViFin for every participant. ViFin still obtained good performance with an average accuracy of numbers at 90% and letters at 89%. These results are summarized in Table 2.

5.5.6 Different Environments. We evaluated ViFin in three scenarios: a quiet office environment (i.e., control group), a noisy office with loud music, and a metro. Unfortunately, we did not get a chance to test ViFin again in a flying airplane due to COVID 19. These three environments are the typical representation of our daily lives in which ViFin could be used as we mentioned in section 3.7. Table 2 enumerates the average accuracy in different environments. The results show invariant accuracy comparing to the control group in a quiet office (90.4% average accuracy). It reveals that ViFin is robust and reliable in different environments.

5.5.7 Temporal Stability. It is crucial to verify that ViFin maintains the temporal stability (i.e., the model of a user is trained once and the resultant system keeps operating in a stable manner over time). The experiments

were conducted spanning over a month at different time periods as shown in Figure 13. The participants were required to write the test set 6 times at 6 different times during a month. The results showed that the accuracy remained constant over one month, which was 85% on average over one month.

5.5.8 System Performance. For the current implementation, the average end-to-end latency is 100 ms from writing the numbers or letters to the display of the input. The initial one round training data set collection lasts for one hour. The initial pre-trained model training process in the cloud takes about 15 hours with 15 participants' data. The fine-tuning takes 15 minutes. The update process takes 2 minutes to update the models. We measure the power consumption of the three models of smartwatch using "Battery Historian" from Google. Specifically, three states are measured: 1) idle with the display on, 2) ViFin with power on but without writing input, and 3) ViFin with power on and continuous writing input. Since the platform is only able to measure the percentage of the battery consumption, we record the time duration for consuming 1% battery for each state. The average resulting time duration in each state is 216 s, 189 s, 178 s, respectively. Given the battery capacity and the working voltage, we calculate the average resulting power consumption of each state, which is 248.2 mW , 284 mW, 298 mW, respectively. Thus, ViFin only consumes an additional 49.8 mW of power on top of the base power consumption. For comparison, we also conduct the measurement when running a step counting application, resulting in the power consumption of 288 mW. Thus, the power consumption of ViFin is similar to the typical application running on a smartwatch.

6 APPLICATIONS AND USER STUDY

We applied ViFin to a wide variety of applications. In this section, we introduce four applications of ViFin, then evaluate two user studies for real-world tasks to further study the system's usability.

6.1 Applications

6.1.1 Equipment Maintenance. Nowadays, engineers in factories are starting to wear smartglasses to maintain and fix machines. While performing their duties, they record information about their machines, such as pieces they have replaced and/or who did it. With a hands-free ViFin system, workers can keep records with ease.

6.1.2 Mixed Reality Education. With the impact of AR/VR technology in education, students may now utilize smartglasses to study 3D structures of buildings and/or biological organs. In addition, with the help of ViFin, students can write notes or answer questions in class. For example, when using a 3D virtual heart model in a biology class, students can point at a section of the heart and label it accordingly.

6.1.3 Home Theatre. ViFin can be connected to smart TV and smartglasses to search movies. For smart TV, users can write a movie name and play the movie without a handhold controller. ViFin can also be connected to smartglasses for movies searching by micro finger writing in the air. In this way, ViFin is more convenient than a handhold controller since the smartglasses block the vision of the controller.

6.1.4 Logistical Management. Stockroom workers are also starting to wear smartglasses to scan barcodes, which is more efficient to transport and record stock than using a handheld device. However, they may encounter difficulties when barcodes do not work since they do not have a touchscreen to input serial numbers manually anymore. Thus, ViFin will solve this problem by allowing employees to use finger writing instead of scanning.

6.2 User Study 1: Standard Transcription Entry

To study the performance of standard transcription entry on a real-world task, we recruited the same five participants mentioned in section 5.5 to write a published phrase set for text entry by MacKenzie and Soukoreff [37]. The standard transcription has 500 phrases. The phases vary from 16 to 43 characters (mean = 28.61). There are

2712 words (1163 unique) varying from 1 to 13 characters (mean = 4.46). The experiment was conducted only with the lower case letters (no upper case letters, punctuation, or numbers). Users were required to have a seat, place their elbow on the table, and write the transcription for two hours.

6.2.1 Speed and Accuracy. Text entry speed was measured in words per minute (wpm). We calculated the number of words the participants wrote in two hours. The average text entry speed for five participants was 16.0 wpm with a standard deviation of 4.12. The average accuracy for two hours was 81.2% with a standard deviation of 2.9. We also calculated the average accuracy for the first hour at 86.5% with a standard deviation of 3.2. We believe that the decreased accuracy in the second hour was because participants got tired and did sloppy writing in the second hour. Note that we did NOT use a spell check or a language model here.

6.3 User Study 2: Movies Searching and Logistical Management

To investigate ViFin's applications in practice, we recruited the same five participants to use ViFin in the applications of movies searching and logistical management, which allowed users to write numbers and letters in the air by micro finger movement. We implemented these two applications on the Mad Gaze smartwatches. One of the applications was a movie player. Users were allowed to search for any movie by writing their names by using their finger in the air with a smartwatch. During writing, the application would display a list of related movies' names with sequence numbers. Then users were allowed to write numbers to choose one from the list and play a movie. In order to test continuous number writing, we designed another application in the smartwatch which allowed user to write and input serial numbers for logistical management. We offered some products with serial numbers and asked users to write using ViFin. In these applications, we use gesture rotation to wake up ViFin. Once Vifin is awake, rotation means switching the input method between numbers and letters. Also, we use left as "backspace" and right as "enter". ViFin sleeps if no finger writing is detected within 20 seconds. For comparison, users were also required to experience these two applications by typing on a phone which connected to the smartglasses. Note that the pictures on the smartglasses partially block the users' vision on the phone screen. Users were allowed to take off the smartglasses when they type on the phone if they want. At last, a questionnaire was distributed.

The questionnaire asked six questions on a scale of 1 to 10, with 1 being the lowest and 10 being the highest.
1. How accurate do you feel the system is when you use ViFin to write with your finger in the air?
2. Do you feel ViFin is more convenient than the phone typing method to search for movies and write numbers on the smartglasses?
3. How tightly do you attach the smartwatch to your wrist to write with finger movement in the air? (10 is very tight)
4. How comfortable do you feel when you experience these two applications?
5. Are you looking forward to using the real ViFin products?
6. How acceptable is it that you have to do the initial data collection process for training before you experience these two applications?

The results shown in Table 3 are quite positive. The average score for questions 1–6 was 9.2, 9.6, 1.8, 10, 10, 8.6 in order.

7 FUTURE WORK

7.1 General Models

Although ViFin improves the accuracy with only one round of training set data (an hour), using pre-training and fine-tuning, which is close to the accuracy of five rounds of the training set (five hours), one-hour initial training is still time-consuming. It will be more practical if ViFin can further reduce the initial training set for new users without affecting accuracy. It is vital to look at the marginal contribution of the new data on improve

Table 3. A questionnaire for user experience

	Q1	Q2	Q3	Q4	Q5	Q6
P1	9	10	2	10	10	8
P2	10	10	2	10	10	9
P3	8	9	1	10	10	7
P4	9	9	1	10	10	9
P5	10	10	3	10	10	10
AVG	9.2	9.6	1.8	10	10	8.6

user-level accuracy and research the use of more advanced transfer learning or other algorithms. We may also not need to collect every combination of letters by performing linguistic analysis because we may further find an excellent trade-off between model accuracy and increased user burden. For example, some speech recognition systems require new users to speak some words or sentences before they use them. Others allow new users to use them directly without training processes. The latter has large speech training sets to train a general model for any user. Fortunately, even though ViFin has inferior accuracy for the general model at this time, we observe that the accuracy improves when the training set increases (see section 5.4). We believe that it is vital to explore a high-performance general model for every user with large training data in the future. It is much more user friendly to make ViFin become a training-free finger writing system for new users. To facilitate this research, we are also planning to collect more training data and make it public.

7.2 Data Labeling

Speech recognition, which already has a high-performance general model, has a large training set while ViFin only has 15 people for training data (5 hours/person), which is 75 hours (825 Mb) in total. It is essential to enlarge the training set if we want to get a high-performance general model for continuous finger writing recognition. However, collecting a big data set is not only challenging to label it, but also time and labor consuming. Base on our experience of ViFin data collection, we found some interesting factors which might affect the quality of training data, such as personality, payment, supervision, and period. Writing for five hours is very boring, and we encountered many participants who quit in the middle of the experiments. We also found that the people who complained a lot during data collection produced very poor quality data with much noise. Interestingly, we found that playing movies during experiments would comfort them and significantly improve the rate of success. But, we do not know whether or how watching movies would distract participants and affect data quality? Also, we found that separating the five-hour experiments into multiple shorter times improves the data quality. Paying another person to supervise experiments improved the data quality a lot. This significantly reduced the noise of extra hand movements during training data collection. However, different supervisors also affected the data quality significantly. For example, some supervisors obtained perfect training data from every participant while some supervisors had very poor quality from every participant, thus we had to replace the latter supervisors. We also found that how much we pay to the participants affects the data quality, which is also related to participants' financial status. Due to the time limitation and COVID 19, we have not identified how these factors exactly affect the data quality and what is the best ways to collect training data. Note that we do not intervene in how participants write. We make a strong effort to get clear and accurate training data, but not test data. Since we want to obtain a large training set, it is vital to itemize the impact of each of these factors and find the most efficient way to collect and label training data in the future.

8 CONCLUSION

In conclusion, we develop a real-time system, ViFin, to recognize **continuous** micro finger writing on a **commodity smartwatch** using vibration for AR/VR interaction. We observe and prove that different finger writing vibrations have unique profiles and can be detected and recognized by a wristband device. ViFin achieves 90% accuracy of continuous numbers and letters recognition across different users by using a GRU_CTC fine-tuned neural network. Five training free gestures are further designed to support extra operations for a text input system, which achieves an accuracy of 100%. A comprehensive series of real-time evaluations and user studies show the effects of different training sizes, calibration, spell checking, different writing speeds, different arm positions, watch displacement, different environments, mobility, gesture control, and how performance varies over 1 month of use. Lastly, we discuss the future work that it is vital to further collect a large amount of data in an efficient way and seek a general model that works for every person.

REFERENCES

[1] [n.d.]. Forward–backward algorithm. https://en.wikipedia.org/wiki/Forward_backward_algorithm.

[2] [n.d.]. How to Write a Spelling Corrector. https://norvig.com/spell-correct.html.

[3] [n.d.]. Network socket. https://en.wikipedia.org/wiki/Network_socket.

[4] [n.d.]. Z-score Nomalization. https://docs.tibco.com/pub/spotfire/6.5.2/doc/html/norm/norm_z_score.htm.

[5] 2017. Leap Motion. https://www.ultraleap.com/.

[6] Heba Abdelnasser, Moustafa Youssef, , and Khaled A Harras. [n.d.]. Wigest: A ubiquitous wifi-based gesture recognition system. In *Proc. IEEE INFOCOM, 2015*.

[7] Liwei Chan, Chi-Hao Hsieh, Yi-Ling Chen, Shuo Yang, Da-Yuan Huang, Rong-Hao Liang, and Bing-Yu Chen. 2015. Cyclops: Wearable and Single-Piece Full-Body Gesture Input Devices. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI EA '15)*. Association for Computing Machinery, New York, NY, USA, 159. https://doi.org/10.1145/2702613.2732482

[8] Ke-Yu Chen, Kent Lyons, Sean White, and Shwetak Patel. 2013. UTrack: 3D Input Using Two Magnetic Sensors. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 237–244. https://doi.org/10.1145/2501988.2502035

[9] Ke-Yu Chen, Shwetak N. Patel, and Sean Keller. 2016. Finexus: Tracking Precise Motions of Multiple Fingertips Using Magnetic Sensing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1504–1514. https://doi.org/10.1145/2858036.2858125

[10] Wenqiang Chen, Lin Chen, Yandao Huang, Xinyu Zhang, Lu Wang, Rukhsana Ruby, and Kaishun Wu. 2019. Taprint: Secure text input for commodity smart wristbands. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[11] Wenqiang Chen, Lin Chen, Kenneth Wan, and John Stankovic. 2020. A smartwatch product provides on-body tapping gestures recognition: demo abstract. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 589–590.

[12] Wenqiang Chen, Maoning Guan, Yandao Huang, Lu Wang, Rukhsana Ruby, Wen Hu, and Kaishun Wu. 2018. Vitype: A cost efficient on-body typing system through vibration. In *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 1–9.

[13] Wenqiang Chen, Maoning Guan, Yandao Huang, Lu Wang, Rukhsana Ruby, Wen Hu, and Kaishun Wu. 2019. A Low Latency On-Body Typing System through Single Vibration Sensor. *IEEE Transactions on Mobile Computing* 19, 11 (2019), 2520–2532.

[14] Wenqiang Chen, Yanming Lian, Lu Wang, Rukhsana Ruby, Wen Hu, and Kaishun Wu. 2017. Virtual keyboard for wearable wristbands. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. 1–2.

[15] Yan Cui, Sebastian Schuon, Derek Chan, Sebastian Thrun, and Christian Theobalt. 2010. 3D shape scanning with a time-of-flight camera. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 1173–1180.

[16] Andrea Giovanni Cutti, Andrea Giovanardi, Laura Rocchi, Angelo Davalli, and Rinaldo Sacchetti. 2008. Ambulatory measurement of shoulder and elbow kinematics through inertial and magnetic sensors. *Med. Bio. Eng. Comput.46, 2, 169–178* (2008).

[17] Jun Gong, Yang Zhang, Xia Zhou, and Xing-Dong Yang. 2017. Pyro: Thumb-Tip Gesture Recognition Using Pyroelectric Infrared Sensing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 553–563. https://doi.org/10.1145/3126594.3126615

[18] Aakar Gupta, Cheng Ji, Hui-Shyong Yeo, Aaron Quigley, and Daniel Vogel. 2019. RotoSwype: Word-Gesture Typing Using a Ring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300244

[19] Sean Gustafson, Christian Holz, and Patrick Baudisch. 2011. Imaginary Phone: Learning Imaginary Interfaces by Transferring Spatial Memory from a Familiar Device. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 283–292. https://doi.org/10.1145/2047196.2047233

[20] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. 2011. OmniTouch: Wearable Multitouch Interaction Everywhere. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 441–450. https://doi.org/10.1145/2047196.2047255

[21] Wenfeng He, Kaishun Wu, Yongpan Zou, and Zhong Ming. [n.d.]. Wig: Wifi-based gesture recognition system. In *Proc. IEEE ICCCN, 2015*.

[22] Jiahui Hou, Xiang-Yang Li, Peide Zhu, Zefan Wang, Yu Wang, Jianwei Qian, and Panlong Yang. [n.d.]. SignSpeaker: A Real-time, High-Precision SmartWatch-based Sign Language Translator. In *Proc. ACM Mobicom, 2019*.

[23] Yi-Ta Hsieh, Antti Jylhä, Valeria Orso, Luciano Gamberini, and Giulio Jacucci. [n.d.]. Designing a Willing-to-Use-in-Public Hand Gestural Interaction Technique for Smart Glasses. In *Proc. ACM CHI, 2016*.

[24] Fang Hu, Peng He, Songlin Xu, Yin Li, and Cheng Zhang. 2020. FingerTrak: Continuous 3D Hand Pose Tracking by Deep Learning Hand Silhouettes Captured by Miniature Thermal Cameras on Wrist. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 2, Article 71 (June 2020), 24 pages. https://doi.org/10.1145/3397306

[25] Da-Yuan Huang, Liwei Chan, Shuo Yang, Fan Wang, Rong-Hao Liang, De-Nian Yan, Yi-Ping Hung, and Bing-Yu Chen. [n.d.]. Designing Thumb-to-Fingers Touch Interfaces for One-Handed and Eyes-Free Interactions. In *Proc. ACM CHI, 2016*.

[26] Yasha Iravantchi, Yang Zhang, Evi Bernitsas, Mayank Goel, and Chris Harrison. 2019. Interferi: Gesture Sensing Using On-Body Acoustic Interferometry. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300506

[27] Daniel Jurafsky and James H. Martin. [n.d.]. Speech and Language Processing.. In *2000*.

[28] Wolf Kienzle and Ken Hinckley. 2014. LightRing: Always-Available 2D Input on Any Surface. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 157–160. https://doi.org/10.1145/2642918.2647376

[29] David Kim, Otmar Hilliges, Shahram Izadi, Alex D. Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. 2012. Digits: Freehand 3D Interactions Anywhere Using a Wrist-Worn Gloveless Sensor. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 167–176. https://doi.org/10.1145/2380116.2380139

[30] Charles H. Knapp and G. Clifford Carter. 1976. The generalized correlation method for estimation of time delay. *IEEE transactions on acoustics, speech, and signal processing* (1976).

[31] Gierad Laput, Robert Xiao, and Chris Harrison. [n.d.]. ViBand: High-Fidelity BioAcoustic Sensing Using Commodity Smartwatch Accelerometers. In *Proc. ACM UIST ,2016*.

[32] DoYoung Lee, SooHwan Lee, and Ian Oakley. 2020. Nailz: Sensing Hand Input with Touch Sensitive Nails. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376778

[33] Lik Hang Lee, Kit Yung Lam, Tong Li, Tristan Braud, Xiang Su, and Pan Hui. 2019. Quadmetric Optimized Thumb-to-Finger Interaction for Force Assisted One-Handed Text Entry on Mobile Headsets. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 94 (Sept. 2019), 27 pages. https://doi.org/10.1145/3351252

[34] Hong Li, Wei Yang, Jianxin Wang, Yang Xu, and Liusheng Huang. [n.d.]. WiFinger: Talk to Your Smart Devices with Finger-grained Gesture. In *Proc. ACM Ubicomp, 2016*.

[35] Jian Liu, Yingying Chen, Marco Grutese, and Yan Wang. [n.d.]. VibSense: Sensing Touches on Ubiquitous Surfaces through Vibration. In *Proc. IEEE Secon, 2017*.

[36] Yongsen Ma, Gang Zhou, Hongyang Zhao Shuangquan Wang, and Woosub Jung. [n.d.]. Signfi: Sign language recognition using wifi. In *Proc. ACM IMWUT, 2018*.

[37] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase Sets for Evaluating Text Entry Techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) *(CHI EA '03)*. Association for Computing Machinery, New York, NY, USA, 754–755. https://doi.org/10.1145/765891.765971

[38] Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. 2017. Real-time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor. In *Proceedings of International Conference on Computer Vision (ICCV)*. 10. https://handtracker.mpi-inf.mpg.de/projects/OccludedHands/

[39] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota1. [n.d.]. Fingerio: Using active sonar for fine-grained finger tracking. In *Proc. ACM CHI, 2016*.

[40] Viet Nguyen, Siddharth Rupavatharam, Luyang Liu, Richard Howard, and Marco Gruteser. [n.d.]. HandSense: Capacitive coupling-based Dynamic, Micro Finger Gesture Recognition. In *Proc. ACM Sensys, 2019*.

[41] Abhinav Parate, Meng-Chieh Chiu, Chaniel Chadowitz, Deepak Ganesan, and Evangelos Kalogerakis. [n.d.]. Risq: Recognizing smoking gestures with inertial sensors on a wristband. In *Proc. ACM International Conference on Mobile Systems, Applications, and Services, 2014.*

[42] Farshid Salemi Parizi, Eric Whitmire, and Shwetak Patel. [n.d.]. AuraRing: Precise Electromagnetic Finger Tracking. In *Proc. ACM IMWUT, 2019.*

[43] Siddharth S. Rautaray and Anupam Agrawal. 2015. Vision Based Hand Gesture Recognition for Human Computer Interaction: A Survey. *Artif. Intell. Rev.* 43, 1 (Jan. 2015), 1–54. https://doi.org/10.1007/s10462-012-9356-9

[44] Hasim Sak, Matt Shannon, Kanishka Rao, and Francoise Beaufays. [n.d.]. Recurrent Neural Aligner: An Encoder-Decoder Neural Network Model for Sequence to Sequence Mapping.. In *Proc. Interspeech, 2017.*

[45] Sheng Shen, He Wang, and Romit Roy Choudhury. [n.d.]. I am a Smartwatch and I can Track my User's Arm. In *Proc. ACM Mobisys, 2016.*

[46] Yilei Shi, Haimo Zhang, Kaixing Zhao, Jiashuo Cao, Mengmeng Sun, and Suranga Nanayakkara. 2020. Ready, Steady, Touch! Sensing Physical Contact with a Finger-Mounted IMU. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 2, Article 59 (June 2020), 25 pages. https://doi.org/10.1145/3397309

[47] Mohamed Soliman, Franziska Mueller, Lena Hegemann, Joan Sol Roo, Christian Theobalt, and Jürgen Steimle. 2018. FingerInput: Capturing Expressive Single-Hand Thumb-to-Finger Microgestures. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (Tokyo, Japan) *(ISS '18)*. Association for Computing Machinery, New York, NY, USA, 177–187. https://doi.org/10.1145/3279778.3279799

[48] Srinath Sridhar, Anders Markussen, Antti Oulasvirta, Christian Theobalt, and Sebastian Boring. 2017. WatchSense: On- and Above-Skin Input Sensing through a Wearable Depth Sensor. In *Proceedings of ACM CHI.* 12. http://handtracker.mpi-inf.mpg.de/projects/WatchSense/

[49] Srinath Sridhar, Franziska Mueller, Antti Oulasvirta, and Christian Theobalt. 2015. Fast and Robust Hand Tracking Using Detection-Guided Optimization. In *Proceedings of Computer Vision and Pattern Recognition (CVPR).* 9. http://handtracker.mpi-inf.mpg.de/projects/FastHandTracker/

[50] Li Sun, Souvik Sen, Dimitrios Koutsonikolas, and Kyu-Han Kim. [n.d.]. Widraw: Enabling hands-free drawing in the air on commodity wifi devices. In *Proc. ACM Mobicom, 2015.*

[51] Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, Arran Topalian, Erroll Wood, Sameh Khamis, Pushmeet Kohli, Toby Sharp, Shahram Izadi, Richard Banks, Andrew Fitzgibbon, and Jamie Shotton. 2016. Efficient and Precise Interactive Hand Tracking through Joint, Continuous Optimization of Pose and Correspondences. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2016* 35 (July 2016). https://www.microsoft.com/en-us/research/publication/efficient-precise-interactive-hand-tracking-joint-continuous-optimization-pose-correspondences/

[52] Jonathan Taylor, Vladimir Tankovich, Danhang Tang, Cem Keskin, David Kim, Philip Davidson, Adarsh Kowdle, and Shahram Izadi. 2017. Articulated Distance Fields for Ultra-Fast Tracking of Hands Interacting. *ACM Trans. Graph.* 36, 6, Article 244 (Nov. 2017), 12 pages. https://doi.org/10.1145/3130800.3130853

[53] Edison Thomaz, Irfan Essa, and Gregory D. Abowd. 2015. A practical approach for recognizing eating moments with wrist-mounted inertial sensing. *ACM International Joint Conference on Pervasive and Ubiquitous Computing, 1029–1040* (2015).

[54] Hoang Truong, Shuo Zhang, Ufuk Muncuk, Phuc Nguyen, Nam Bui, Anh Nguyen, Qin Lv, Kaushik Chowdhury, Thang Dinh, and Tam Vu. 2018. CapBand: Battery-Free Successive Capacitance Sensing Wristband for Hand Gesture Recognition. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems* (Shenzhen, China) *(SenSys '18)*. Association for Computing Machinery, New York, NY, USA, 54–67. https://doi.org/10.1145/3274783.3274854

[55] Robert Wang, Sylvain Paris, and Jovan Popoviundefined. 2011. 6D Hands: Markerless Hand-Tracking for Computer Aided Design. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 549–558. https://doi.org/10.1145/2047196.2047269

[56] Saiwen Wang, Jie Song, Jaime Lien, Ivan Poupyrev, and Otmar Hilliges. 2016. Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 851–860. https://doi.org/10.1145/2984511.2984565

[57] Wei Wang, Alex X Liu, and Ke Sun. 2016. Device-free gesture tracking using acoustic signals. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking.* 82–94.

[58] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. [n.d.]. Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. In *Proc. ACM CHI ,2016.*

[59] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. [n.d.]. Digitouch: Reconfigurable thumb-to-finger input and text entry on head-mounted displays. In *Proc. ACM IMWUT, 2017.*

[60] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. 2017. DigiTouch: Reconfigurable Thumb-to-Finger Input and Text Entry on Head-Mounted Displays. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 113 (Sept. 2017), 21 pages. https://doi.org/10.1145/3130978

[61] Mathias Wilhelm, Daniel Krakowczyk, and Sahin Albayrak. 2020. PeriSense: Ring-Based Multi-Finger Gesture Interaction Utilizing Capacitive Proximity Sensing. *Sensors* 20 (07 2020), 3990. https://doi.org/10.3390/s20143990

[62] Kaishun Wu, Yandao Huang, Wenqiang Chen, Lin Chen, Xinyu Zhang, Lu Wang, and Rukhsana Ruby. 2020. Power saving and secure text input for commodity smart watches. *IEEE Transactions on Mobile Computing* (2020).

[63] Chao Xu, Parth H. Pathak, and Prasant Mohapatra. [n.d.]. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proc. ACM HotMobile, 2015.*

[64] Xing-Dong Yang, Tovi Grossman, Daniel Wigdor, and George Fitzmaurice. 2012. Magic Finger: Always-Available Input through Finger Instrumentation. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 147–156. https://doi.org/10.1145/2380116.2380137

[65] Hui-Shyong Yeo, Juyoung Lee, Hyung-il Kim, Aakar Gupta, Andrea Bianchi, Daniel Vogel, Hideki Koike, Woontack Woo, and Aaron Quigley. 2019. WRIST: Watch-Ring Interaction and Sensing Technique for Wrist Gestures and Macro-Micro Pointing. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 19, 15 pages. https://doi.org/10.1145/3338286.3340130

[66] Hui-Shyong Yeo, Erwin Wu, Juyoung Lee, Aaron Quigley, and Hideki Koike. 2019. Opisthenar: Hand Poses and Finger Tapping Recognition by Observing Back of Hand Using Embedded Wrist Camera. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 963–971. https://doi.org/10.1145/3332165.3347867

[67] Yafeng Yin, Lei Xie, Tao Gu, Yijia Lu, and Sanglu Lu. 2019. AirContour: Building Contour-based Model for In-Air Writing Gesture Recognition. *ACM Transactions on Sensor Networks, Vol. 15, No. 4, Article 44* (2019).

[68] Sang Ho Yoon, Ke Huo, Vinh P. Nguyen, and Karthik Ramani. [n.d.]. TIMMi: Finger-worn Textile Input Device with Multimodal Sensing in Mobile Interaction. In *Proc. ACM TEI, 2015.*

[69] Sang Ho Yoon, Ke Huo, and Karthik Ramani. [n.d.]. Plex: Finger-worn Textile Sensor for Mobile Interaction During Activities.. In *Proc. ACM UbiComp, 2014.*

[70] Yang Zhang and Chris Harrison. 2015. Tomo: Wearable, Low-Cost Electrical Impedance Tomography for Hand Gesture Recognition. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 167–173. https://doi.org/10.1145/2807442.2807480

[71] Huiyu Zhou, Huosheng Hu, and Yaqin Tao. 2006. Inertial measurements of upper limb motion. *Med. Bio. Eng. Comput.44, 6, 479–487.* (2006).

A APPENDIX

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	0.97	0	0	0	0.01	0	0.01	0	0	0	0	0	0.01	0.01	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0.01	0	0	0.98	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0
E	0	0	0.01	0	0.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0.01
F	0	0	0	0.01	0	0.95	0	0	0	0	0	0.02	0	0.01	0	0	0	0.01	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0.97	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0.01	0.01
H	0	0.01	0	0	0	0	0	0.98	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0
I	0.01	0	0	0	0.01	0	0	0	0.98	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0.01	0	0.99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	0	0.97	0	0	0	0	0.01	0.01	0.01	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0.01	0	0.01	0	0.01	0	0.95	0	0	0	0	0	0	0	0	0	0.01	0	0	0.01	0.01
M	0	0	0	0	0	0	0	0	0	0	0	0	0.97	0.03	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0.02	0	0	0	0.01	0.01	0.89	0	0	0.01	0.01	0.01	0	0	0.03	0.02	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.96	0	0.01	0	0.03	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0.01	0.01	0	0	0	0	0.99	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0.99	0	0	0	0	0	0	0	0	0
R	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0.99	0	0	0	0	0	0	0	0.01
S	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0.99	0	0	0	0	0	0	0
T	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0.98	0	0	0	0	0	0.01
U	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0.94	0.01	0.05	0	0	0
V	0	0.01	0	0	0	0	0	0	0	0	0	0.01	0	0.01	0	0	0	0	0	0	0.01	0.95	0	0.01	0	0
W	0.01	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.97	0	0	0.01
X	0	0	0.01	0	0	0	0	0	0	0.01	0	0	0	0	0.01	0.01	0	0	0	0	0	0.01	0	0.95	0.01	0
Y	0	0	0	0.01	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0.97	0
Z	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.99

Fig. 14. Confusion matrix of letters